1. Give an example where call apply bind is required?

call() - This method is used to borrow a function from an object and use that function with the data of another object.

Example:

```
let name={
        firstname:"Ganesh",
        lastname:"Prabhu"
          }
Let printname=function(hometown,state){
console.log(this.firstname+this.lastname+"from"+hometown+state)
          }

printname.call(name,"bhatkal","karnatak");
              |      |_______|
             Object       |
             Name    arguments
```

apply() - This method is the same as a call but the arguments are stored in an array manner

Example: printname.apply(name,["bhatkal","karnatak"]);

bind() - In the bind method it will take the copy of the function which we are invoking and it will return that to a variable and it can be called whenever the function is required.

Example :

```
Let printthename=printname.bind(name,"bhatkal","karnatak");
 printthename();
```

The call and apply methods are called the functions wherever they are called but the bind function copies that function and saves it to the variable.

2. What is the difference between readFile and readFileSync?

Readfile	ReadfileSync
1. It has a non-blocking nature.	1. It has a blocking nature.
2. It uses a callback function.	2. It uses a variable to store the file contents.
3. Due to its non-blocking nature it won't wait for the file to be read completely.	3. Due to blocking nature it will wait till the file is read completely.

3. What does process in node.js mean?

The process object in Node.js is a global object that can be accessed inside any module without requiring it. It is an essential component in the Node.js ecosystem as it provides various information sets about the runtime of a program(current nodejs process).

4. Explain what node.js is?

Node.js (Node) is an open-source development platform for executing JavaScript code server-side. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds, and web push notifications.

5. What is the difference between JS from browser to JS on node.js

In the browser, most of the time what you are doing is interacting with the DOM or other Web Platform APIs like Cookies.
Those do not exist in Node.js, of course. You don't have the document, window, and all the other objects that are provided by the browser.

Another big difference is that in Node.js you control the environment. Unless you are building an open-source application that anyone can deploy anywhere, you know which version of Node.js you will run the application on.
The browser environment, where you don't get the luxury to choose what browser your visitors will use, is very convenient.

Another difference is that Node.js supports both the CommonJS and ES module systems (since Node.js v12), while in the browser we are starting to see the ES Modules standard being implemented.
In practice, this means that you can use both require() and import in Node.js, while you are limited to import in the browser.

6. Write three different ways to reverse a string in Javascript? a. using an inbuilt method, b. iteratively, c. recursively

a. **INBUILT METHOD**

```
Let str=”abc”
  str.split("").reverse().join("");
console.log(str)    //output : cba
```

B. ITERATIVELY

```
Let str=”abc”,newstr=””
 for(let i=str.length-1;i>=0;i- -){
     newstr+=str[i]
}
console.log(newstr)     // output  :  cba
```

c.Recursively

```
 function reverseString(str) {
  if (str === "")
    return "";
  else
    return reverseString(str.substr(1)) + str.charAt(0);
}
reverseString("hello");      //output  :hello
```

7. Write a program to check two objects are equal (deep equal)

```
function deepEqual(object1, object2) {
 const keys1 = Object.keys(object1);
 const keys2 = Object.keys(object2);
 if (keys1.length !== keys2.length) {
   return false;
 }
 for (const key of keys1) {
   const val1 = object1[key];
   const val2 = object2[key];
   const areObjects = isObject(val1) && isObject(val2);
   if (
     areObjects && !deepEqual(val1, val2) ||
     !areObjects && val1 !== val2
   ) {
     return false;
   }
 }
 return true;
}
function isObject(object) {
```

```
 return object != null && typeof object === 'object';   }
```

8. What is shallow equal?

During shallow equality check of objects you get the list of properties (using Object.keys()) of both objects, then check the properties' values for equality.

```
      function shallowEqual(object1, object2) {
 const keys1 = Object.keys(object1);
 const keys2 = Object.keys(object2);
 if (keys1.length !== keys2.length) {
   return false;
 }
 for (let key of keys1) {
   if (object1[key] !== object2[key]) {
     return false;
   }
 }
 return true;
}
```